

# Solving Reachability Problems with Singular and Indefinite Hessian by Sequential Quadratic Programming

Jan Kuřátko<sup>\*†</sup>

November 4, 2016

## Abstract

The problem of finding a solution of a dynamical system that originates and terminates in given sets of states is considered. We formulate it as an equality constrained nonlinear programming problem and compare line search and trust-region approaches to solving it. We put special emphasis on studying properties of the Hessian matrix which results from the optimization problem. Results concerning the spectrum and the structure of the Hessian are given. Formulas for the computation of the Hessian are listed and the consequences of various approximation schemes for the Hessian are demonstrated on benchmarks.

## 1 Introduction

In this paper we are concerned with a task of finding a solution to a dynamical system that starts in a given set and reaches another set of states. We assume those two sets to be ellipsoids and call them the set of initial states and the set of unsafe states. The problem at hand is not a classical boundary value problem [9] and we solve it by formulating it as an equality constrained nonlinear programming problem. To this end we apply the *sequential quadratic programming* method (SQP) [13, Ch. 18].

We present an analysis of the resulting optimization problem. We study the structure and properties of the Hessian matrix in more detail. In particular we are interested in the spectrum of the matrix. We discuss which solution approaches are most suitable with respect to the properties of the Hessian and the whole saddle-point matrix. In addition we do computational experiments both line-search SQP [13, Alg. 18.3] and trust-region SQP [13, Alg. 18.4] on benchmark problems.

The motivation for this work stems from the field of computer aided verification [5, 10, 16]. Here a solution from the set of initial states that reaches the set of unsafe states may represent a flaw in the design of a system. To this end researchers try to develop automatized methods for identifying such flaws [2]. There are several approaches to this problem [1, 9, 16].

In the previous paper [9] several formulations of the objective function and vector of constraints were considered that were solved by SQP. In this paper we study in more detail the setup that performed best. In addition we provide additional theoretical as well as computational results. The main contributions are:

- formulas for the Hessian matrix and its use,

---

<sup>\*</sup>ORCID: 0000-0003-0105-6527; Faculty of Mathematics and Physics, Charles University, Czech Republic; Institute of Computer Science, The Czech Academy of Sciences

<sup>†</sup>This work was supported by the Czech Science Foundation (GACR) grant number 15-14484S with institutional support RVO:67985807.

- analysis of the spectrum of the Hessian,
- approximations of the Hessian and reasons for its ill-conditioning,
- comparison of line-search SQP with trust-region SQP,
- discussion about stopping criteria.

The structure of the presented paper is the following. We formulate the problem we try to solve and introduce the notation in Section 2. In Section 3 we review SQP and show the structure of the Hessian matrix. In addition we prove some properties regarding the spectrum of the Hessian. The solution of the saddle-point matrix as well as the choice of the step length in the optimization process is left for Sections 4 and 5. Then there follow computational experiments in Section 6 and the paper finishes with a summary of results.

## 2 Problem Formulation

Consider a dynamical system whose dynamics is governed by a system of ordinary differential equations of the following form

$$\dot{x} = f(x(t)), \quad (1)$$

where  $x : \mathbb{R} \rightarrow \mathbb{R}^n$  is an unknown function of a variable  $t \geq 0$ , the right hand side  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is continuously differentiable. For the *Flow* function  $\Phi : \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^n$ , when one fixes the initial value  $x_0$ , then one obtains the solution  $x(t)$  of (1). Therefore, for the initial time  $t = 0$  we have  $\Phi(0, x_0) = x_0$ , and for  $t \geq 0$ ,  $\Phi(t, x_0) = x(t)$ . We try to solve the following classical problem from the field of computer aided verification [5, 10, 16].

**Problem.** *Consider a dynamical system (1) and let Init and Unsafe be two sets of states in  $\mathbb{R}^n$ . Find a solution of (1) such that the initial state  $x_0 \in \text{Init}$  and  $\Phi(t, x_0) \in \text{Unsafe}$  for some  $t \geq 0$ .*

At this point we consider both Init and Unsafe to be ellipsoids. The resulting boundary value problem is underdetermined with an unknown upper bound on time  $t \geq 0$ .

We reformulate and try to solve this problem as a minimization problem where one seeks a solution to

$$\min F(\chi) \quad \text{subject to} \quad c(\chi) = 0. \quad (2)$$

We use the idea of multiple shooting [3, Ch. 4] where one finds a solution on a longer time interval by patching up a number of solution segments on shorter time intervals. One such a solution to our problem stated above is illustrated in Fig. 1. Here the unknown vector  $\chi$  consists of the initial states of solution segments  $x_0^i \in \mathbb{R}^n$  and lengths of segments  $t_i \geq 0$ ,  $1 \leq i \leq N$ , and has the following form

$$\chi = [x_0^1, t_1, x_0^2, t_2, \dots, x_0^N, t_N]^T \in \mathbb{R}^{N(n+1)}. \quad (3)$$

The objective function  $F(\chi)$  and vector of constraints  $c(\chi)$  is chosen such that

$$F(\chi) = \sum_{i=1}^N t_i^2, \quad (4)$$

$$c(\chi) = [g_I, g_1, \dots, g_{N-1}, g_U] \in \mathbb{R}^{(N-1)n+2}, \quad (5)$$

where  $g_I = \frac{1}{2} (\|x_0^1 - c_I\|_{E_I}^2 - 1) \in \mathbb{R}$ ,  $g_U = \frac{1}{2} (\|\Phi(t_N, x_0^N) - c_U\|_{E_U}^2 - 1) \in \mathbb{R}$ , and constraints  $g_i = x_0^{i+1} - \Phi(t_i, x_0^i) \in \mathbb{R}^n$  for  $1 \leq i \leq N-1$ . Here  $\|\cdot\|_{E_I}$  and  $\|\cdot\|_{E_U}$  are energy norms induced by symmetric positive definite (SPD) matrices  $E_I \in \mathbb{R}^{n \times n}$ ,  $E_U \in \mathbb{R}^{n \times n}$

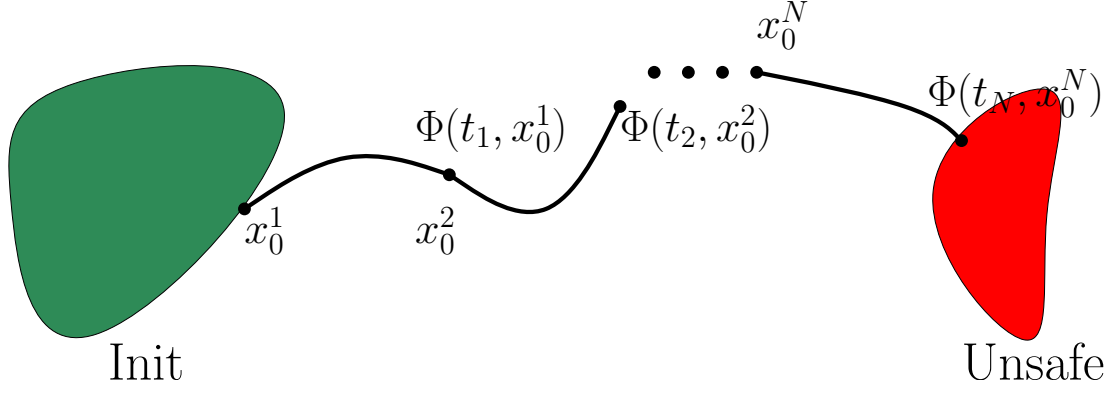


Figure 1: Connected  $N$  solution segments

respectively. Since the sets Init and Unsafe are ellipsoids their shape and size is given by  $E_I$  and  $E_U$ .

The constraints  $g_i$  are so called matching conditions enforcing two consecutive segments to be connected. The two constraints  $g_I$  and  $g_U$  force  $x_0^1$  to be on the boundary of Init and  $\Phi(t_N, x_0^N)$  to be on the boundary of Unsafe, provided  $c_I$  and  $c_U$  are centres of ellipsoids.

To solve (2) one can use Sequential Quadratic Programming (SQP) [13, Ch. 18]. The choice of  $F(\chi)$  and  $c(\chi)$  is a result of [9] where we conducted a number of experiments with various formulations for the minimization problem (2). Although in our numerical experiments this choice of the objective function and the vector of constraints worked best [9], there were remaining problems with the Hessian approximation and conditioning of the resulting saddle point matrix.

In the following we provide deeper analysis of sources of numerical problems stemming from the use of approximation schemes for the Hessian. In addition to work [9] we address the use of indefinite Hessians in trust-region methods and compare it to the line-search approach.

### 3 KKT Matrix

When one solves (2) by SQP a Lagrangian  $\mathcal{L}(\chi, \lambda)$  is formed, where  $\lambda \in \mathbb{R}^{(N-1)n+2}$  is a vector of Lagrange multipliers such that

$$\lambda = [\lambda_I, \lambda_1, \dots, \lambda_{N-1}, \lambda_U]^T \in \mathbb{R}^{(N-1)n+2} \quad (6)$$

with  $\lambda_I \in \mathbb{R}$ ,  $\lambda_U \in \mathbb{R}$  and  $\lambda_i \in \mathbb{R}^n$  for  $1 \leq i \leq N-1$ . Let  $B(\chi) \in \mathbb{R}^{N(n+1) \times (N-1)n+2}$  be the Jacobian of the vector of constraints such that  $B = [\nabla_\chi g_I, \nabla_\chi g_1, \dots, \nabla_\chi g_{N-1}, \nabla_\chi g_U]$ . Then for the Lagrangian  $\mathcal{L}(\chi, \lambda) = F(\chi) + \lambda^T c(\chi)$  one gets

$$\nabla_\chi \mathcal{L}(\chi, \lambda) = \nabla_\chi F(\chi) + B(\chi) \lambda, \quad (7)$$

$$\nabla_\lambda \mathcal{L}(\chi, \lambda) = c(\chi). \quad (8)$$

The resulting saddle point system one needs to solve in every iteration [13, Ch. 18] is

$$\begin{bmatrix} H & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} d_\chi \\ d_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla_\chi \mathcal{L}(\chi, \lambda) \\ -\nabla_\lambda \mathcal{L}(\chi, \lambda) \end{bmatrix}, \quad (9)$$

which we write shortly as

$$Kd = b,$$

where  $H \in \mathbb{R}^{N(n+1) \times N(n+1)}$  is either  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$  or its approximation. The solution vector of (9) is then used to compute the next iterate

$$\chi^+ = \chi + \alpha_\chi d_\chi \text{ and } \lambda^+ = \lambda + \alpha_\lambda d_\lambda, \quad (10)$$

where  $\alpha_\chi = \alpha_\lambda = 1$  gives the Newton method to the nonlinear KKT system (7)-(8). Note that because of the structure of (4) and (5) the matrix  $H$  is a block diagonal matrix such that

$$H = \begin{bmatrix} H_1 & & \\ & \ddots & \\ & & H_N \end{bmatrix} \in \mathbb{R}^{N(n+1) \times N(n+1)}, \quad (11)$$

with the block  $H_i \in \mathbb{R}^{(n+1) \times (n+1)}$  for  $1 \leq i \leq N$ .

### 3.1 Structure

Denote the sensitivity function of the solution  $x(t) = [x_1(t), \dots, x_n(t)]^T$  of the differential equation to the change of the initial value  $x_0 = [x_{0,1}, \dots, x_{0,n}]^T \in \mathbb{R}^n$  by  $S : \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$ , and denote the Jacobian of the right-hand side of (1) by  $\partial f(x(t))/\partial x$ . We use the convention

$$S(t, x_0) = \frac{\partial \Phi(t, x_0)}{\partial x_0} = \begin{bmatrix} \frac{\partial x_1(t)}{\partial x_{0,1}} & \dots & \frac{\partial x_1(t)}{\partial x_{0,n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_n(t)}{\partial x_{0,1}} & \dots & \frac{\partial x_n(t)}{\partial x_{0,n}} \end{bmatrix}, \quad \frac{\partial f(x(t))}{\partial x} = \begin{bmatrix} \frac{\partial f_1(x(t))}{\partial x_1} & \dots & \frac{\partial f_1(x(t))}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n(x(t))}{\partial x_1} & \dots & \frac{\partial f_n(x(t))}{\partial x_n} \end{bmatrix}, \quad (12)$$

where  $f(x(t)) = [f_1(x(t)), \dots, f_n(x(t))]^T \in \mathbb{R}^n$ . Then [9] the Jacobian of constraint is given by

$$B(\chi) = \begin{bmatrix} v_I & -M_1^T & & & & \\ & -v_1^T & & & & \\ & I & -M_2^T & & & \\ & & -v_2^T & & & \\ & & I & & & \\ & & & \ddots & & \\ & & & & -M_{N-1}^T & \\ & & & & -v_{N-1}^T & \\ & & & & I & v_U \\ & & & & 0 & \beta_U \end{bmatrix} = \begin{bmatrix} \square & & & & & \\ - & & & & & \\ & \square & & & & \\ & - & & & & \\ & & \square & & & \\ & & - & & & \\ & & & \ddots & & \\ & & & & \square & \\ & & & & - & \\ & & & & & \square \\ & & & & & - \\ & & & & & \circ \end{bmatrix}, \quad (13)$$

where

$$\begin{aligned} v_I &= E_I(x_0^1 - c_I) \in \mathbb{R}^n, \\ v_U &= S(t_N, x_0^N)^T E_U(\Phi(t_N, x_0^N) - c_U) \in \mathbb{R}^n, \\ \beta_U &= \frac{d\Phi(t_N, x_0^N)}{dt_N}^T E_U(\Phi(t_N, x_0^N) - c_U) \in \mathbb{R}, \end{aligned}$$

and for,  $1 \leq i \leq N-1$ ,  $M_i = S(t_i, x_0^i) \in \mathbb{R}^{n \times n}$  and  $v_i = d\Phi(t_i, x_0^i)/dt_i \in \mathbb{R}^n$ . Visualization of the structure of  $B(\chi)$  is shown on the right hand side of (13).

The following lemmas describe the structure of the Hessian  $H$  depending on the dynamics of (1).

**Lemma 1.** *Consider the dynamical system (1) and the minimization problem (2) with the objective function given by (4) and vector of constraints given by (5). Then the Hessian of the Lagrangian  $\mathcal{L}(\chi, \lambda)$  is a block diagonal matrix of the form*

$$\nabla_\chi^2 \mathcal{L}(\chi, \lambda) = \begin{bmatrix} \begin{bmatrix} A_1 & v_1 \\ v_1^T & \alpha_1 \end{bmatrix} & & \\ & \ddots & \\ & & \begin{bmatrix} A_N & v_N \\ v_N^T & \alpha_N \end{bmatrix} \end{bmatrix} \in \mathbb{R}^{N(n+1) \times N(n+1)},$$

where  $n$  is the statespace dimension, and  $N$  is the number of segments. For  $1 \leq i \leq N$ , blocks  $A_i \in \mathbb{R}^{n \times n}$ ,  $v_i \in \mathbb{R}^n$  and  $\alpha_i \in \mathbb{R}$ . For Lagrange multipliers  $\lambda_i \in \mathbb{R}^n$ ,  $1 \leq i \leq N-1$  we have

$$\begin{aligned} v_i &= - \left( \frac{\partial f(x(t_i))}{\partial x} S(t_i, x_0^i) \right)^T \lambda_i, \quad 1 \leq i \leq N-1, \\ \alpha_i &= - \left( \frac{\partial f(x(t_i))}{\partial x} f(x(t_i)) \right)^T \lambda_i + 1, \quad 1 \leq i \leq N-1, \\ A_1 &= \lambda_1 E_1 - \frac{\partial^2 \Phi(t_1, x_0^1)}{\partial x_0^{12}} \circ \lambda_1, \\ A_i &= - \frac{\partial^2 \Phi(t_i, x_0^i)}{\partial x_0^{i2}} \circ \lambda_i, \quad 2 \leq i \leq N-1, \end{aligned}$$

where  $\frac{\partial^2 \Phi(t_i, x_0^i)}{\partial x_0^{i2}} \in \mathbb{R}^{n \times n \times n}$  is a tensor and the symbol  $\circ$  denotes the contraction by  $\lambda_i$ . The last block of  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$  consists of

$$\begin{aligned} A_N &= \lambda_U S(t_N, x_0^N)^T E_U S(t_N, x_0^N) + \lambda_U \frac{\partial^2 \Phi(t_N, x_0^N)}{\partial x_0^{N2}} \circ E_U (\Phi(t_N, x_0^N) - c_U), \\ v_N &= \lambda_U \left( \frac{\partial f(x(t_N))}{\partial x} S(t_N, x_0^N) \right)^T E_U (\Phi(t_N, x_0^N) - c_U) + \lambda_U S(t_N, x_0^N)^T E_U f(x(t_N)), \\ \alpha_N &= \lambda_U \left( \frac{\partial f(x(t_N))}{\partial x} f(x(t_N)) \right)^T E_U (\Phi(t_N, x_0^N) - c_U) + \lambda_U f(x(t_N))^T E_U f(x(t_N)) + 1, \end{aligned}$$

where  $\frac{\partial^2 \Phi(t_N, x_0^N)}{\partial x_0^{N2}} \in \mathbb{R}^{n \times n \times n}$  is a tensor.

*Proof.* The proof follows from differentiating  $\mathcal{L}(\chi, \lambda)$  twice with respect to parameter  $\chi$ . First, one gets  $\nabla_\chi \mathcal{L}(\chi, \lambda) = \nabla_\chi F(\chi) + B(\chi)\lambda$ , and then

$$\begin{aligned} \nabla_\chi^2 \mathcal{L}(\chi, \lambda) &= \nabla_\chi^2 F(\chi) + \sum \lambda_i \circ \nabla_\chi^2 g_i(\chi) \\ &= \nabla_\chi^2 F(\chi) + \cdots \\ &\quad \cdots + \lambda_I \nabla_\chi^2 g_I(\chi) + \lambda_1 \circ \nabla_\chi^2 g_1(\chi) + \cdots + \lambda_{N-1} \circ \nabla_\chi^2 g_{N-1}(\chi) + \lambda_U \nabla_\chi^2 g_U(\chi). \end{aligned}$$

The term  $\nabla_\chi^2 F(\chi) \in \mathbb{R}^{N(n+1) \times N(n+1)}$  is a diagonal matrix containing the second derivatives with respect to  $t_i$ ,  $1 \leq i \leq N$ , of the term  $\frac{1}{2} \sum_i^N t_i^2$ . Therefore, there are only  $N$  nonzero elements  $d^2 F(\chi)/dt_i^2$ ,  $1 \leq i \leq N$ . Those are the “+1” in formulas for  $\alpha_i$ ,  $1 \leq i \leq N$ .

Let us have a look at the matching conditions  $g_i(\chi) = x_0^{i+1} - \Phi(t_i, x_0^i)$ ,  $1 \leq i \leq N-1$ , and observe that those are dependent on  $x_0^{i+1}$ ,  $x_0^i$  and  $t_i$ . The term  $g_i(\chi)$  vanishes when differentiated twice with respect to  $x_0^{i+1}$ . Therefore  $\lambda_i \circ \nabla_\chi^2 g_i(\chi) \in \mathbb{R}^{N(n+1) \times N(n+1)}$  features possibly nonzero entries only in elements corresponding to the second and mixed derivatives of  $x_0^i$  and  $t_i$ ,  $1 \leq i \leq N-1$ . Hence one obtains the block-diagonal structure of the Hessian.

To compute the second mixed derivatives first put  $d\Phi(t_i, x_0^i)/dt = f(x(t_i)) \in \mathbb{R}^n$  and then  $\partial f(x(t_i))/\partial x_0^i = \partial f(x(t_i))/\partial x \cdot S(t_i, x_0^i) \in \mathbb{R}^{n \times n}$ . Those are contained in formulas for  $v_i$ ,  $1 \leq i \leq N-1$ . The second derivative with respect to  $t$ , using the chain rule, is then  $d^2 \Phi(t_i, x_0^i)/dt^2 = df(x(t_i))/dt = \partial f(x(t_i))/\partial x \cdot f(x(t_i)) \in \mathbb{R}^n$ . These formulas you can find in  $\alpha_i$ ,  $1 \leq i \leq N-1$ .

In the case of the first constraint  $g_I(\chi) = 0.5 ((x_0^1 - c_I)^T E_I (x_0^1 - c_I) - 1)$ , its second derivative with respect to  $x_0^1$  is  $E_I \in \mathbb{R}^{n \times n}$ . The computation is more difficult for the constraint  $g_U(\chi) = 0.5 ((\Phi(t_N, x_0^N) - c_U)^T E_U (\Phi(t_N, x_0^N) - c_U) - 1)$ , however,  $A_N$ ,  $v_N$

and  $\alpha_N$  is again the result of the application of the chain rule. The first derivatives with respect to  $x_0^N$  and  $t_N$  are

$$\frac{\partial g_U(\chi)}{\partial x_0^N} = S(t_N, x_0^N)^T E_U(\Phi(t_N, x_0^N) - c_U), \quad \frac{dg_U(\chi)}{dt_N} = f(x(t_N))^T E_U(\Phi(t_N, x_0^N) - c_U).$$

Differentiating both terms again with respect to  $x_0^N$  and  $t_N$  delivers desired formulas for  $A_N$ ,  $v_N$  and  $\alpha_N$ .  $\square$

The *Flow* of a linear ODE  $\dot{x} = Ax(t)$  is  $\Phi(t, x_0) = e^{At}x_0$ , where  $x_0$  is an initial state and  $t \geq 0$ . Then  $S(t, x_0) = e^{At}$ , the second derivative with respect to  $x_0$  vanishes, and  $f(x(t)) = Ae^{At}x_0$ . From this one obtains the following.

**Lemma 2.** *Consider a linear dynamical system  $\dot{x} = Ax(t)$  and the minimization problem (2) with the objective function given by (4) and vector of constraints given by (5). Then the Hessian of the Lagrangian  $\mathcal{L}(\chi, \lambda)$  is a block diagonal matrix of the form*

$$\nabla_{\chi}^2 \mathcal{L}(\chi, \lambda) = \begin{bmatrix} \begin{bmatrix} A_1 & v_1 \\ v_1^T & \alpha_1 \end{bmatrix} & & \\ & \ddots & \\ & & \begin{bmatrix} A_N & v_N \\ v_N^T & \alpha_N \end{bmatrix} \end{bmatrix} \in \mathbb{R}^{N(n+1) \times N(n+1)},$$

where  $n$  is the statespace dimension, and  $N$  is the number of segments. For  $1 \leq i \leq N$ , blocks  $A_i \in \mathbb{R}^{n \times n}$ ,  $v_i \in \mathbb{R}^n$  and  $\alpha_i \in \mathbb{R}$ . For Lagrange multipliers  $\lambda_i \in \mathbb{R}^n$ ,  $1 \leq i \leq N-1$  we have

$$\begin{aligned} v_i &= -(Ae^{At_i})^T \lambda_i, \quad 1 \leq i \leq N-1, \\ \alpha_i &= -(A^2 e^{At_i} x_0^i)^T \lambda_i + 1, \quad 1 \leq i \leq N-1, \\ A_1 &= \lambda_1 E_1, \\ A_i &= 0, \quad 2 \leq i \leq N-1, \\ A_N &= \lambda_U e^{A^T t_N} E_U e^{At_N}, \\ v_N &= \lambda_U (Ae^{At_N})^T E_U (e^{At_N} x_0^N - c_U) + \lambda_U e^{A^T t_N} E_U A e^{At_N} x_0^N, \\ \alpha_N &= \lambda_U (A^2 e^{At_N} x_0^N)^T E_U (e^{At_N} x_0^N - c_U) + \lambda_U (Ae^{At_N} x_0^N)^T E_U A e^{At_N} x_0^N + 1. \end{aligned}$$

*Proof.* The result follows directly from Lemma 1.  $\square$

Lemmas 1 and 2 show the block diagonal structure of the Hessian matrix  $H$ . In addition, when the dynamics in (1) is linear, one does not need to differentiate the Lagrangian  $\mathcal{L}(\chi, \lambda)$  twice with respect to  $\chi$ . Computed data from the Jacobian (13), in particular sensitivity functions  $S(t_i, x_0^i) = e^{At_i}$ ,  $1 \leq i \leq N-1$ , can be used in the formulas of Lemma 2. Because of this observation one can work with the Hessian given by analytical formulas with no extra computational effort.

As for the nonlinear dynamics (1) there are tensors  $\partial^2 \Phi(t_i, x_0^i) / \partial x_0^{i,2} \in \mathbb{R}^{n \times n \times n}$  for  $2 \leq i \leq N$  in the formulas of Lemma 1. Because of this we cannot get  $H$  without additional computation. Either we use numerical differentiation or use an approximation scheme for the Hessian. However, it is possible to compute one part of the Hessian exactly using formulas for  $v_i$  and  $\alpha_i$ ,  $1 \leq i \leq N$ , and the rest by some approximation scheme [7].

## 3.2 Properties

We are interested in the spectrum of  $H$  and the whole saddle point matrix as well as the conditions on solvability of the KKT system (9). Since there is a complete description of  $H$  in Lemma 2, let us start there.

**Lemma 3.** Let  $M \in \mathbb{R}^{(n+1) \times (n+1)}$  be a matrix of the form

$$M = \begin{bmatrix} 0 & v \\ v^T & \alpha \end{bmatrix},$$

where the upper-left block  $0 \in \mathbb{R}^{n \times n}$ ,  $v \in \mathbb{R}^n$  and  $\alpha \in \mathbb{R}$ . Then  $M$  has  $(n-1)$  zero eigenvalues and additional two such that

$$\lambda_{\pm} = \frac{\alpha \pm \sqrt{\alpha^2 + 4v^T v}}{2}.$$

*Proof.* Eigenvalues and eigenvectors  $(\lambda, u)$  of  $M$  satisfy  $Mu = \lambda u$  with  $u = [x^T, y^T]^T$  where

$$\begin{bmatrix} 0 & v \\ v^T & \alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} x \\ y \end{bmatrix}.$$

First assume that  $\lambda = 0$ . Then one can find  $(n-1)$  orthogonal eigenvectors  $u = [x^T, 0]^T$  satisfying  $v^T x = 0$  since  $v \in \mathbb{R}^n$ .

By rewriting the matrix equation above one gets

$$\begin{aligned} vy &= \lambda x \\ v^T x + \alpha y &= \lambda y. \end{aligned}$$

When  $\lambda \neq 0$  then  $y$  needs to be nonzero. It follows from the first equation that for  $y = 0$  one gets  $x = 0$ . Dividing the first equation by  $\lambda$  and substituting for  $x$  in the second, one obtains  $v^T vy + \lambda \alpha y = \lambda^2 y$ . Since  $y \neq 0$  we can divide both sides and put  $\lambda^2 - \lambda \alpha - v^T v = 0$ .  $\square$

**Lemma 4.** Let  $M \in \mathbb{R}^{(n+1) \times (n+1)}$  be a matrix of the form

$$M = \begin{bmatrix} A & v \\ v^T & \alpha \end{bmatrix},$$

where  $A \in \mathbb{R}^{n \times n}$  is SPD (SND),  $v \in \mathbb{R}^n$  and  $\alpha \in \mathbb{R}$ . Then  $M$  has  $n$  strictly positive (negative, respectively) eigenvalues and  $\lambda_{n+1} = \alpha - v^T A^{-1} v$ .

*Proof.* Since  $A$  is a SPD (SND) matrix then matrix  $M$  can be factorized in the following way [4, Sec. 3.4]

$$M = \begin{bmatrix} I & 0 \\ v^T A^{-1} & 1 \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & \alpha - v^T A^{-1} v \end{bmatrix} \begin{bmatrix} I & A^{-1} v \\ 0 & 1 \end{bmatrix}.$$

Therefore the eigenvalues of  $M$  are eigenvalues of  $A$  and  $\lambda_{n+1} = \alpha - v^T A^{-1} v$ .  $\square$

Lemmas 3 and 4 tell us that for linear ODEs the Hessian matrix is singular with both positive and negative eigenvalues. Moreover, the dimension of the nullspace of  $H$  is at least  $(N-2)(n-1)$ , where  $n$  is the state space dimension and  $N$  is the number of solution segments.

As discussed in [6] higher nullity of  $H$  than  $(N-1)n+2$ , assuming  $B(\chi)$  has full column rank, implies that the saddle point matrix (9) is singular. Under an additional assumption on  $v_i$  and  $\alpha_i$  over Lemma 2 we are able to conclude that the maximum nullity of  $H$  is less than  $(N-1)n+2$ .

**Lemma 5.** Let  $H \in \mathbb{R}^{N(n+1) \times N(n+1)}$  be a block diagonal matrix with blocks  $H_i \in \mathbb{R}^{(n+1) \times (n+1)}$   $1 \leq i \leq N$ . Let each block be of the form

$$H_i = \begin{bmatrix} A_i & v_i \\ v_i^T & \alpha_i \end{bmatrix}.$$

Let  $A_1 \in \mathbb{R}^{n \times n}$  and  $A_N \in \mathbb{R}^{n \times n}$  be each either SPD or SND matrix. Let  $A_i \in \mathbb{R}^{n \times n}$ ,  $2 \leq i \leq N-1$ , be zero matrices. Moreover, assume that at least one of  $v_i$  and  $\alpha_i$ ,  $2 \leq i \leq N-1$ , in each block is nonzero. Then the maximum dimension of the null-space of  $H$  is  $(N-2)n+2$ .

*Proof.* One can count the possible maximum number of zero eigenvalues of  $H$ . Using Lemma 4 it follows that there are at most one zero eigenvalue in each block  $H_1$  and  $H_N$ . In the remaining blocks  $H_i$ ,  $2 \leq i \leq N-1$ , one gets the most zero eigenvalues when  $v_i = 0$ ,  $2 \leq i \leq N-1$ . It follows from Lemma 3 that we get additional  $(N-2)n$  zero eigenvalues.  $\square$

For nonlinear ODEs we do not have similar results to those in Lemmas 3, 4 and 5. In our experiments when we used numerical differentiation to compute the Hessian  $H$  it happened to be always an indefinite matrix.

## 4 Solution of the KKT System

Finding the minimum of (2) by SQP requires computation of the solution of (9) in every iteration. There are three choices: either one solves the original linear system, or uses the Schur-complement method, or the Null space method [4, 12]. In this paper we assume the saddle point matrix to be nonsingular.

Each method has its advantages and disadvantages in dependence on the properties and structure of the saddle point matrix. In our problem we deal with singular and indefinite Hessian blocks  $H$  as shown by Lemmas 1 and 2. Because of this we avoid using Schur complement methods since they require the existence of  $H^{-1}$  [4, Sec. 5].

Therefore, in our implementation of line-search SQP we use the Null space method, in particular, the preconditioned projected conjugate gradient method [12, Alg. NPCG]. Here, one uses an indefinite (constraint) preconditioner [4, Sec. 10.2]

$$C = \begin{bmatrix} D & B \\ B^T & 0 \end{bmatrix}, \quad (14)$$

where  $B$  is the Jacobian of constraints  $c(\chi)$  and  $D \in \mathbb{R}^{N(n+1) \times N(n+1)}$  is SPD. One chooses the matrix  $D$  so that it is simple and the linear system with the coefficient matrix  $B^T D^{-1} B$  can be solved efficiently. Often leaving  $D = I$  is enough [4, p. 81].

**Lemma 6.** *Let  $B$  be the Jacobian of the vector of constraints (13). Denote by*

$$\begin{aligned} v_I &= E_I(x_0^1 - c_I), \\ v_U &= S(t_N, x_0^N)^T E_U(\Phi(t_N, x_0^N) - c_U), \\ \alpha &= \frac{d\Phi(t_N, x_0^N)^T}{dt_N} E_U(\Phi(t_N, x_0^N) - c_U), \end{aligned}$$

and for  $1 \leq i \leq N-1$  put

$$\begin{aligned} M_i &= -S(t_i, x_0^i)^T, \\ v_i &= -\frac{d\Phi(t_i, x_0^i)}{dt_i}, \\ D_i &= M_i^T M_i + v_i v_i^T + I. \end{aligned}$$

Then

$$B^T B = \begin{bmatrix} v_I^T v_I & v_I^T M_1 & & & & \\ M_1^T v_I & D_1 & M_2 & & & \\ & M_2^T & D_2 & & & \\ & & & \ddots & & \\ & & & & D_{N-2} & M_{N-1} \\ & & & & M_{N-1}^T & D_{N-1} & v_U \\ & & & & & v_U^T & v_U^T v_U + \alpha^2 \end{bmatrix}. \quad (15)$$



*Proof.* The formula for  $B^T B$  follows from (13) by direct multiplication.  $\square$

Lemma 6 shows that the matrix  $B^T B$  is banded and there is no fill-in after matrix multiplication. Therefore, the Cholesky factors of  $B^T B$  are also not dense matrices and have banded structure as well. The width of the band in (15) is independent of the number of solution segments  $N$ . For the reader's convenience, in Alg. 1 we present an instantiation of *NPCG* with  $D = I$  and our notation.

---

**Algorithm 1** *NPCG* [12, p. 8] with  $D = I$  for line-search SQP

---

```

 $\omega \leftarrow 0.9$ 
 $d_\chi \leftarrow -B(B^T B)^{-1} \nabla_\lambda \mathcal{L}(\chi, \lambda)$ 
 $r_z \leftarrow -\nabla_\chi \mathcal{L}(\chi, \lambda) - H d_\chi$ 
 $d_\lambda \leftarrow (B^T B)^{-1} B^T r_z$ 
 $\tilde{r}_z \leftarrow (r_z - B d_\lambda)$ 
 $\gamma \leftarrow r_z^T \tilde{r}_z$ 
 $\bar{\gamma} \leftarrow \gamma$ 
 $p_z \leftarrow \tilde{r}_z$ 
while  $\gamma > \omega \bar{\gamma}$  do
   $q_z \leftarrow H p_z$ 
   $\alpha \leftarrow \gamma / (p_z^T q_z)$ 
   $d_\chi^+ \leftarrow d_\chi + \alpha p_z$ 
   $r_z^+ \leftarrow r_z - \alpha q_z$ 
   $d_\lambda^+ \leftarrow (B^T B)^{-1} B^T r_z^+$ 
   $\tilde{r}_z^+ \leftarrow (r_z^+ - B d_\lambda^+)$ 
   $\gamma^+ \leftarrow (r_z^+)^T \tilde{r}_z^+$ 
   $\beta \leftarrow \gamma^+ / \gamma$ 
   $p_z^+ \leftarrow \tilde{r}_z^+ + \beta p_z$ 
   $(d_\chi, d_\lambda, r_z, \tilde{r}_z, p_z, \gamma) \leftarrow (d_\chi^+, d_\lambda^+, r_z^+, \tilde{r}_z^+, p_z^+, \gamma^+)$ 
end while
Output:  $d_\chi$  and  $d_\lambda$ 

```

---

## 5 Step Selection

The Newton method applied to the nonlinear KKT system (7)-(8) with  $\alpha_\chi = \alpha_\lambda = 1$  is locally convergent. If  $\chi$  and  $\lambda$  are not close to the solution  $\chi^*$  and  $\lambda^*$  of (7)-(8), then the Newton method can fail. Therefore a globalization strategy for computing the step length  $\alpha$  is necessary [12].

We also address computation and approximation of the Hessian matrix. A popular way of approximating the Hessian  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$  is to use Quasi-Newton methods [13, Chap. 6]. One such a method is called *BFGS* and it produces SPD approximations to the Hessian [13, Sec. 6.1]. Another popular Quasi-Newton method is SR-1 [13, Sec. 6.2] that produces an indefinite approximation of the Hessian. In addition one may use finite differences [13, Ch. 8] to compute  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$ .

There are two basic approaches to the computation of solution vector  $d$  (9) and step length  $\alpha > 0$ : line-search methods and trust-region methods [13]. Let us investigate and compare their behaviour.

### 5.1 Line-search Methods

Line-search methods require either the upper-left block  $H$  of (9) to be an SPD approximation of the Hessian  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$  or the projection of the Hessian onto the null space

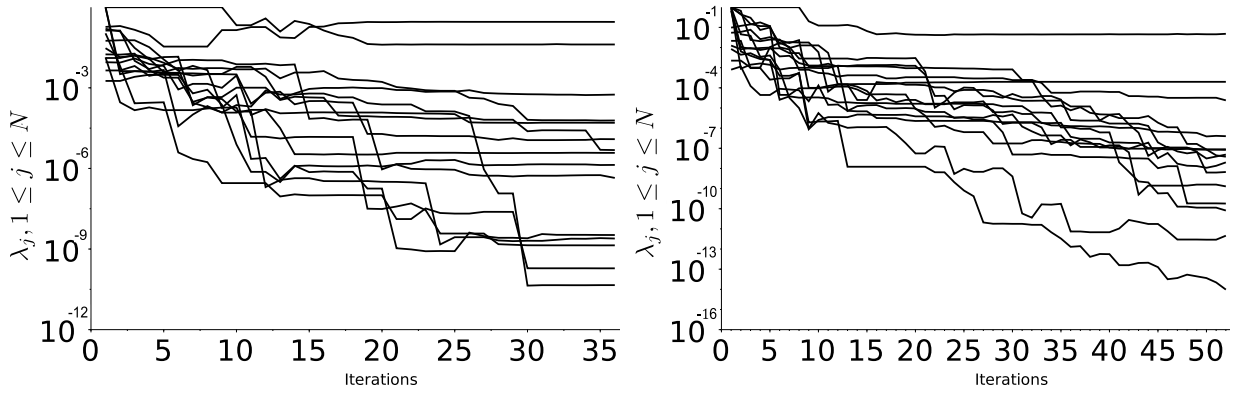


Figure 2: Values of the minimal eigenvalue in each block of  $H$ . State space dimension is  $n = 20$ , number of segments is  $N = 15$ . The ODEs 6.2 on the left hand side are linear, the ODEs 6.3 on the right hand side are nonlinear.

of  $B^T(\chi)$  to be an SPD matrix [13, Sec. 18.4]. When this requirement is fulfilled then the solution  $d_\chi$  is a descent direction. However, it may happen that  $d_\chi$  is not acceptable because it increases the violation of the vector of constraints [13, Sec. 15.4]. When this happens one can reset the Hessian to a diagonal SPD matrix and compute a new  $d_\chi$ . In section 6 we provide a rule for measuring a sufficient decrease in the value of the objective function and satisfaction of constraints.

Convergence properties of variable metric matrices generated by BFGS were studied in [14]. In our problem we try to solve the KKT system (9) that features singular and indefinite Hessian  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$ . To our knowledge, there are no results showing that the sequence of variable metric matrices generated by BFGS converges when  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$  is not an SPD matrix.

Let us illustrate a typical behaviour when BFGS is used block-wise [7], [13, Sec. 7.3-7.4] for approximating  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$ . We draw the value of the minimal eigenvalue in each block  $H_i$ ,  $1 \leq i \leq N$ , of  $H$  in Fig. 2. One can see that BFGS approximation features eigenvalues close to zero. This is a source of ill-conditioning of  $H$  and the whole KKT system (9) as shown in [4, Th. 3.5]. Although now  $H^{-1}$  exists and it is even an SPD matrix, one should again avoid using Schur complement methods because of the ill-conditioning of  $H$ .

The suitable step size  $\alpha > 0$  can then be computed using a merit function [12, Sec. 3]. In our implementation we use [12,  $P_2(\alpha)$ ] that is

$$P(\alpha) = F(\chi + \alpha d_\chi) + (\lambda + d_\lambda)^T c(\chi + \alpha d_\chi) + \frac{\sigma}{2} \|c(\chi + \alpha d_\chi)\|^2. \quad (16)$$

Properties of (16) are described in [12, Th. 8]. For stepsize selection and termination *Backtracking Line Search* [13, Alg. 3.1] can be used. In our implementation the stepsize selection is terminated whenever  $P(\alpha) - P(0) \leq \delta \alpha P'(0)$  for  $\delta = 10^{-4}$ . If the inequality is not satisfied one needs to reduce  $\alpha$ . Our strategy is to divide  $\alpha$  by 2 and try again.

## 5.2 Trust-region Methods

An alternative approach to line-search methods are trust-region methods. Instead of computing a direction  $(d_\chi, d_\lambda)$  first and then adjusting the step size  $\alpha > 0$ , trust region method chooses a radius  $\Delta > 0$  first and then a suitable direction  $(d_\chi, d_\lambda)$  [13, Ch. 4]. Trust-region SQP methods are attractive because they do not require  $\nabla_\chi^2 \mathcal{L}(\chi, \lambda)$  and its approximation to be an SPD matrix [13, Sec. 18.5].

Two essential parts of the method are the computation of the radius  $\Delta$  and of  $(d_\chi, d_\lambda)$ .

We monitor the ratio

$$\begin{aligned}\varrho &= \frac{P(1) - P(0)}{Q(1) - Q(0)}, \\ &= \frac{d_\chi^T (\nabla_\chi F(\chi) + B(\chi)(\lambda + d_\lambda)) + \sigma d_\chi^T B(\chi)c(\chi)}{d_\chi^T (\nabla_\chi F(\chi) + B(\chi)(\lambda + d_\lambda)) + \frac{1}{2}d_\chi^T H d_\chi + \sigma d_\chi^T B(\chi)c(\chi)},\end{aligned}$$

in order to judge the acceptability of  $d_\chi$ . Note that the merit function  $P(\alpha)$  is given by (16) and the choice of  $\sigma$  follows results in [11, Th. 10]. Then the computation of the trust-region radius is described in Alg. 2, which follows [13, Alg. 4.1]. We tried different constants for deciding on the acceptability of  $d_\chi$ , such as  $\varrho < 0.1$  and  $\varrho > 0.9$  in Alg. 2, however, it did not lead to any improvement. In addition, increasing the maximum radius  $\hat{\Delta}$  to 10 did not affect the computation.

---

**Algorithm 2** TR-Radius [13, p. 69]

---

```

 $\hat{\Delta} \leftarrow 1.5$ 
 $\Delta$  from previous iteration
 $\varrho \leftarrow \frac{P(1)-P(0)}{Q(1)-Q(0)}$ 
if  $\varrho < \frac{1}{4}$  then
     $\Delta^+ \leftarrow \frac{1}{4}\Delta$ 
else if  $\varrho > \frac{3}{4}$  and  $\|d_\chi\| = \Delta$  then
     $\Delta^+ \leftarrow \min(2\Delta, \hat{\Delta})$ 
else
     $\Delta^+ \leftarrow \Delta$ 
end if
if  $\varrho > 0$  then
     $\chi^+ \leftarrow \chi + d_\chi$ 
     $\lambda^+ \leftarrow \lambda + d_\lambda$ 
else
     $\chi^+ \leftarrow \chi$ 
     $\lambda^+ \leftarrow \lambda$ 
end if
 $(\chi, \lambda, \Delta) \leftarrow (\chi^+, \lambda^+, \Delta^+)$ 
Output:  $\chi$ ,  $\lambda$  and  $\Delta$ 

```

---

In order to compute  $d_\chi$  and  $d_\lambda$  one can follow trust-region SQP [13, Alg. 18.4]. For repeatability purposes and for the reader's convenience we present [11, Alg. *TRCG*] which is one instantiation of key parts of trust-region SQP [13, Alg. 18.4]. The result [11, Alg. *TRCG*] is presented using our notation in Alg. 3. Note that we fixed a typo  $d_\chi < \Delta$  and  $d_\chi + \alpha p_z < \Delta$ , and a problem with not exiting a while loop, and we added a detection of negative curvature of the projected Hessian on the null-space of  $B^T(\chi)$  as used in Steihaug-Toint algorithm.

Note that there is a possibility of a breakdown in both Alg. 1 and 3. In the case of exact Hessian  $H$  for linear ODEs it may happen that  $H p_z = 0$ , since  $H$  is singular. In such a case one can restart the method and put  $H = D$ , where  $D$  is an SPD diagonal matrix. If  $p_z^T H p_z < 0$ , then it only causes problems in Alg. 1. In our experience this happens rarely which confirms findings in [12, p. 263].

## 6 Computational Experiments

We used the Line search and the Trust-region methods described in the paper on a series of benchmarks. Key parts of each method are described in more detail in Alg. 1 for the

---

**Algorithm 3** *TRCG* [11, p. 14]

$\vartheta \leftarrow 4/5, \omega \leftarrow 0.9$  ▷ Compute Newton and Cauchy steps [13, p. 547-548].  
 $d_\chi^C \leftarrow -(\|Bc\|^2/\|B^T Bc\|^2) Bc$   
 $d_\chi^N \leftarrow -B(B^T B)^{-1} c$   
**if**  $\|d_\chi^C\| \geq \vartheta\Delta$  **then** ▷ Use Dog-leg method [13, Sec. 4.1].  
     $d_\chi \leftarrow (\vartheta\Delta/\|d_\chi^C\|) d_\chi^C$   
**else if**  $\|d_\chi^C\| < \vartheta\Delta < \|d_\chi^N\|$  **then**  
     $\alpha \leftarrow -(d_\chi^C)^T(d_\chi^N - d_\chi^C) + \sqrt{((d_\chi^C)^T(d_\chi^N - d_\chi^C))^2 + (\vartheta\Delta)^2 - \|d_\chi^C\|^2}$   
     $d_\chi \leftarrow d_\chi^C + \alpha(d_\chi^N - d_\chi^C)$   
**else**  
     $d_\chi \leftarrow d_\chi^N$   
**end if** ▷ Apply the preconditioned projected conjugate gradient method.  
 $r_z \leftarrow -\nabla_\chi \mathcal{L}(\chi, \lambda) - Hd_\chi$   
 $d_\lambda \leftarrow (B^T B)^{-1} B^T r_z$   
 $\tilde{r}_z \leftarrow r_z - Bd_\lambda$   
 $\gamma \leftarrow r_z^T \tilde{r}_z$   
 $\bar{\gamma} \leftarrow \gamma$   
 $p_z \leftarrow \tilde{r}_z$   
**while**  $\gamma > \omega\bar{\gamma}$  and  $\|d_\chi\| < \Delta$  **do**  
     $q_z \leftarrow Hp_z$   
     $\alpha \leftarrow \gamma/(p_z^T q_z)$   
    **if**  $\|d_\chi + \alpha p_z\| < \Delta$  and  $\alpha > 0$  **then**  
         $d_\chi^+ \leftarrow d_\chi + \alpha p_z$   
         $r_z^+ \leftarrow r_z - \alpha q_z$   
         $d_\lambda^+ \leftarrow (B^T B)^{-1} B^T r_z^+$   
         $\tilde{r}_z^+ \leftarrow r_z^+ - Hd_\lambda^+$   
         $\gamma^+ \leftarrow (r_z^+)^T \tilde{r}_z^+$   
         $\beta \leftarrow \gamma^+/\gamma$   
         $p_z^+ \leftarrow \tilde{r}_z^+ + \beta p_z$   
    **else** ▷ If the curvature of the Hessian is negative or  $\|d_\chi + \alpha p_z\| > \Delta$ , then set  $\alpha$  so that  $\|d_\chi^+\| = \Delta$  and stop.  
         $\alpha \leftarrow -p_z^T d_\chi + \sqrt{(p_z^T d_\chi)^2 + \Delta^2 - \|d_\chi\|^2}$   
         $d_\chi^+ \leftarrow d_\chi + \alpha p_z$   
         $r_z^+ \leftarrow r_z - \alpha q_z$   
         $d_\lambda^+ \leftarrow (B^T B)^{-1} B^T r_z^+$   
        exit while loop  
    **end if**  
     $(d_\chi, d_\lambda, r_z, \tilde{r}_z, p_z, \gamma) \leftarrow (d_\chi^+, d_\lambda^+, r_z^+, \tilde{r}_z^+, p_z^+, \gamma^+)$   
**end while**  
Output:  $d_\chi$  and  $d_\lambda$

---

the line search and Alg. 2, 3 for the trust-region approach.

For computing and approximating  $\nabla_{\chi}^2 \mathcal{L}(\chi, \lambda)$  we tried different possibilities such as *BFGS*, *SR-1* and second derivatives computed by finite differences. When the true Hessian or its approximation is not an SPD matrix, then line-search may produce a  $d_{\chi}$  that is not a descent direction. In our implementation, we reject the solution vector  $d_{\chi}$  of the saddle-point system (9) whenever  $-P'(0) < 10^{-5} \|d_{\chi}\| \|\nabla_{\chi} \mathcal{L}(\chi, \lambda)\|$  [12, Alg. 3.1]. In that case we restart the method setting the Hessian to be the identity matrix.

We are interested in how well those methods scale with respect to the state space dimension  $n$  and the number of segments  $N$ . In addition we compare line search and trust-region method.

That is for the given dynamical system (1) and state space dimension  $n$  we set  $c_I = [1, \dots, 1]^T \in \mathbb{R}^n$  to be the initial state. Then we compute  $c_U = \Phi(5, c_I)$ . The sets *Init* and *Unsafe* are balls of radius  $1/4$  centred at  $c_I$  and  $c_U$  respectively. We create  $N$  solution segments by splitting the solution segment from  $c_I$  to  $c_U$  such that each segment has length  $t = 5/N$ . Denote the initial states by  $x_0^i$ ,  $1 \leq i \leq N$ , and modify them by  $x_0^i + u$ , where  $u = 0.5 \times [-1, 1, \dots, (-1)^n]^T \in \mathbb{R}^n$ . With these updated initial conditions and lengths  $t_i = 5/N$  we get a vector of parameters (3). From these  $N$  segments we try to compute a solution with  $x_0^1 \in \text{Init}$  and  $\Phi(t_N, x_0^N) \in \text{Unsafe}$ .

We use the following *stopping criteria*: optimality and feasibility conditions are  $\|\nabla_{\chi} \mathcal{L}(\chi, \lambda)\| < 10^{-3}$  and  $\|\nabla_{\lambda} \mathcal{L}(\chi, \lambda)\| < 10^{-8}$  both satisfied (S 1); maximum number of iterations is 400 (S 2); the step size  $\alpha < 10^{-8}$  for line search and the radius  $\Delta < 10^{-8}$  for trust-region (S 3); If any of these stopping criteria are met then the method terminates. In the end we verify by simulation that parameters (3) give the desired solution. The solution vector  $\chi$  is said to be verified by simulation if  $(x_0^1 - c_I)^T E_I (x_0^1 - c_I) < 1 + \varepsilon$  and  $(\Phi(\sum t_i, x_0^1) - c_U)^T E_U (\Phi(\sum t_i, x_0^1) - c_U) < 1 + \varepsilon$ , where  $\varepsilon = 10^{-4}$ . In case they do not it is marked by “F”. In tables with results there are rows and columns marked by an “S”. In those it is written which stopping criterion took place and values range from 1 to 3 as explained above with failed attempts marked by “F”. Finally, the number of iterations is denoted by *NIT*.

All computations were carried out in *Scilab* 5.5.2 [15] installed on a computer with *Cent OS* 6.8. In our implementation, one iteration in line search SQP takes similar amount of running time as one iteration in trust-region SQP. Therefore, we only list the number of iterations in the tables with results.

## 6.1 Benchmark 1

Consider the following nonlinear dynamical system

$$\begin{aligned}\dot{x}_1 &= -x_2 + x_1 x_3, \\ \dot{x}_2 &= x_1 + x_2 x_3, \\ \dot{x}_3 &= -x_3 - (x_1^2 + x_2^2) + x_3^3,\end{aligned}$$

that is adopted from [8, p. 334]. In addition we compare what approximation scheme for the Hessian may be the most suited.

In Tab. 1 there are results when the Hessian blocks  $H_i$ ,  $1 \leq i \leq N$ , are approximated by BFGS. When we used SR-1 we got results in Tab. 2. Note that in cases where the maximum number of iterations was reached we got feasibility conditions  $\|\nabla_{\lambda} \mathcal{L}(\chi, \lambda)\| \leq 10^{-7}$  except for  $N = 25$  where  $\|\nabla_{\lambda} \mathcal{L}(\chi, \lambda)\| \leq 10^{-4}$ . For the results in Tab. 3 a built-in function *numderivative* was used for computing the Hessian. Both approaches yield similar results in terms of the number of iterations in this case.

Let us go back to the results in Tab. 2 where we used SR-1 method to approximate the Hessian. We list the number of restarts in the line-search method when we needed to change the approximated Hessian for the identity matrix to get a descent direction  $d_{\chi}$ . With increasing  $N$  we have: 1, 0, 2, 4, 0 and 2 restarts.

N	5	10	15	20	25	30	N	5	10	15	20	25	30
NIT	35	33	31	48	47	49	NIT	28	32	36	43	45	51
S	1	1	1	1	3	3	S	1	1	1	1	1	1

Table 1: Benchmark 1 with the Hessian matrix approximated by BFGS; In the left there are results for line search, and in the right there are results for trust region.

N	5	10	15	20	25	30	N	5	10	15	20	25	30
NIT	16	28	34	46	7	47	NIT	29	400	127	400	400	400
S	1	1	3	3	F	3	S	1	2	1	2	2	2

Table 2: Benchmark 1 with the Hessian matrix approximated by SR1; In the left there are results for line search, and in the right there are results for trust region.

## 6.2 Benchmark 2

Consider the following linear dynamical system

$$\dot{x} = Ax$$

$$= \begin{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} & & & \\ & \ddots & & \\ & & \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} & \\ & & & \end{bmatrix} x,$$

where  $A \in \mathbb{R}^{n \times n}$ . This benchmark problem can not only be scaled up in the number of solution segments  $N$  but as well as in the state space dimension  $n$ . When  $n = 40$  and  $N = 30$  we solve constrained optimization problem with  $N(n + 1) = 1230$  parameters.

Note that the Hessian  $H \in \mathbb{R}^{N(n+1) \times N(n+1)}$ . Therefore, it is important to keep the block diagonal structure 2 and approximate (compute) only  $H_i$ ,  $1 \leq i \leq N$ . Then the Hessian has up to  $N(n + 1)^2$  nonzero elements and we keep the sparse structure.

In Tab. 4 there are results when the BFGS method was used. One can see that the line search method outperforms the trust-region method, especially, for the higher values of  $n$ . The results for SR-1 approximation scheme are shown in Tab. 5. Note that, when SR-1 was used, the number of restarts in line search method for  $n = 40$  and  $N = 5$  was zero. However, for the setting  $n = 40$  and  $N = 28$  we needed to reset the Hessian to be the identity matrix twenty-eight times.

When the formulas from Lemma 2 are used one gets the results in Tab. 6. Notice that the trust-region approach almost always terminates because of the maximum number of iterations condition. However, when we investigate the feasibility condition we get that  $\|\nabla_\lambda \mathcal{L}(\chi, \lambda)\| \leq 10^{-4}$  in all of those cases. In particular for  $n = 40$  and  $N = 10$  we have  $\|\nabla_\chi \mathcal{L}(\chi, \lambda)\| \leq 10^{-2}$  and  $\|\nabla_\lambda \mathcal{L}(\chi, \lambda)\| \leq 10^{-12}$ . It is interesting to see how the feasibility conditions behave through iterations for two settings of  $n$  and  $N$  as illustrated in Fig. 3 when the trust-region approach is used. Note that we have an acceptable result way before the 400th iteration.

N	5	10	15	20	25	30	N	5	10	15	20	25	30
NIT	29	41	20	25	31	24	NIT	24	44	21	27	46	36
S	1	1	3	3	3	3	S	1	1	1	1	1	1

Table 3: Benchmark 1 with the Hessian matrix computed by *numderivative*; In the left there are results for line search, and in the right there are results for trust region.

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	28	1	20	5	33	1	30	5	30	1	40	5	34	1
	10	31	1		10	39	1		10	172	1		10	29	1
	15	400	2		15	36	1		15	39	1		15	37	1
	20	48	1		20	41	1		20	172	1		20	32	1
	25	42	1		25	36	1		25	108	1		25	87	3
	30	35	1		30	39	1		30	44	1		30	51	1

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	26	1	20	5	28	1	30	5	29	1	40	5	30	1
	10	91	1		10	400	2		10	400	2		10	400	2
	15	119	1		15	90	1		15	308	1		15	400	2
	20	400	2		20	101	1		20	400	2		20	400	2
	25	38	1		25	95	1		25	400	2		25	400	2
	30	39	1		30	53	1		30	112	1		30	311	1

Table 4: Benchmark 2 with the Hessian matrix approximated by BFGS; In the top there are results for line search, and in the bottom there are results for trust region.

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	64	1	20	5	35	1	30	5	39	1	40	5	27	1
	10	37	1		10	34	1		10	35	1		10	34	1
	15	53	1		15	70	1		15	50	1		15	62	1
	20	38	1		20	34	1		20	58	1		20	45	1
	25	52	1		25	41	1		25	45	1		25	128	1
	30	54	1		30	46	1		30	100	1		30	57	1

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	400	2	20	5	400	2	30	5	400	F	40	5	400	F
	10	400	2		10	400	2		10	400	2		10	400	2
	15	400	2		15	400	2		15	400	2		15	400	2
	20	400	F		20	400	2		20	400	F		20	400	F
	25	400	F		25	400	2		25	400	2		25	400	2
	30	400	2		30	400	2		30	400	2		30	400	2

Table 5: Benchmark 2 with the Hessian matrix approximated by SR1; In the top there are results for line search, and in the bottom there are results for trust region.

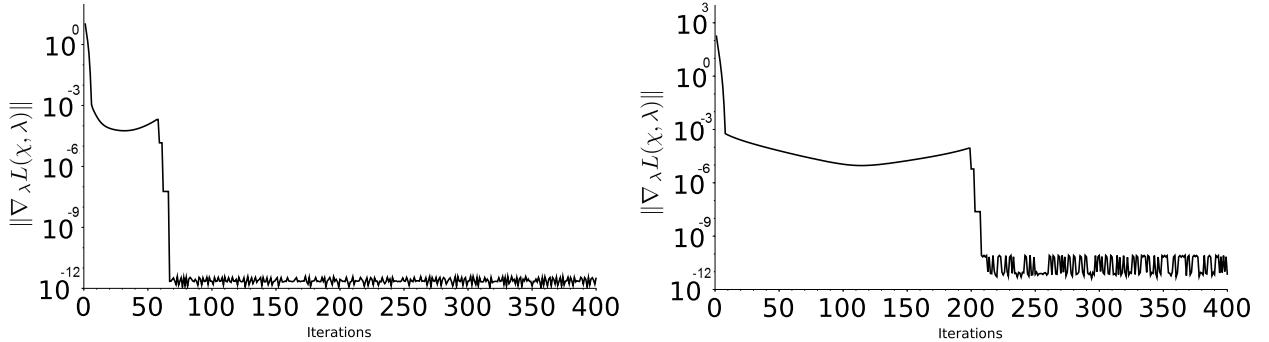


Figure 3: The progress of feasibility conditions of trust-region SQP of Benchmark 2 with the exact Hessian: the setting is  $n = 10$  and  $N = 25$  in the left;  $n = 40$  and  $N = 10$  in the right.

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	43	1	20	5	47	1	30	5	27	1	40	5	26	1
	10	117	1		10	108	1		10	121	1		10	23	1
	15	49	1		15	32	1		15	76	1		15	41	1
	20	47	1		20	23	3		20	48	1		20	30	1
	25	29	1		25	49	1		25	39	1		25	31	1
	30	32	1		30	34	1		30	34	1		30	33	1

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	185	1	20	5	28	1	30	5	400	2	40	5	400	2
	10	400	2		10	400	2		10	400	2		10	400	2
	15	247	1		15	400	2		15	400	2		15	400	2
	20	400	F		20	400	2		20	400	2		20	400	F
	25	400	2		25	400	2		25	400	2		25	400	2
	30	400	2		30	400	2		30	400	2		30	400	2

Table 6: Benchmark 2 with the exact Hessian; In the top there are results for line search, and in the bottom there are results for trust region.

### 6.3 Benchmark 3

Consider the following nonlinear dynamical system

$$\dot{x} = Ax + \sin(x^r)$$

$$= \begin{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} & & \\ & \ddots & \\ & & \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \end{bmatrix} x + \begin{bmatrix} \sin(x_n) \\ \vdots \\ \sin(x_1) \end{bmatrix},$$

where  $A \in \mathbb{R}^n$ . It is similar to benchmark 6.2, however, this time there is a nonlinear term  $\sin(x^r)$  present. This causes that blocks  $A_i$ ,  $1 \leq i \leq N$ , in the Hessian 1 to be nonzero in general.

One can see in Tab. 7 that both approaches yield similar results when BFGS was used. All runs terminated successfully with no fails. As for the SR-1 approximation of the Hessian  $H$  the results in Tab. 8 are not that promising. Both approaches failed to find an acceptable solution in few cases.

Note that in Tab. 8 the trust-region method always used the maximum number of iterations. The reason behind is that the norm  $\|\nabla_{\chi} \mathcal{L}(\chi, \lambda)\|$  does not drop below the prescribed tolerance  $10^{-3}$ . Let us visualize how  $\|\nabla_{\chi} \mathcal{L}(\chi, \lambda)\|$  and  $\|\nabla_{\lambda} \mathcal{L}(\chi, \lambda)\|$  change over iterations for  $n = 40$  and  $N = 20$  in Fig. 4. In addition, from our experience it follows that the number of restarts in line search does not tell us whether our method converges to a desired solution. For instance, when SR-1 was used, there was only one restart for  $n = 10$  and  $N = 25$ , yet our method failed to find a solution. On the other hand our method found a solution for  $n = 40$  and  $N = 25$  although it had to reset the Hessian for the identity matrix eighteen times.

## 7 Conclusion

In this paper we investigated the problem of finding a solution to a dynamical system that starts in a given set and reaches another set of states. We studied properties of the saddle-point matrix (9) resulting from a minimization formulation (2) of this problem. In addition, we compared line search and trust-region methods on benchmark problems.



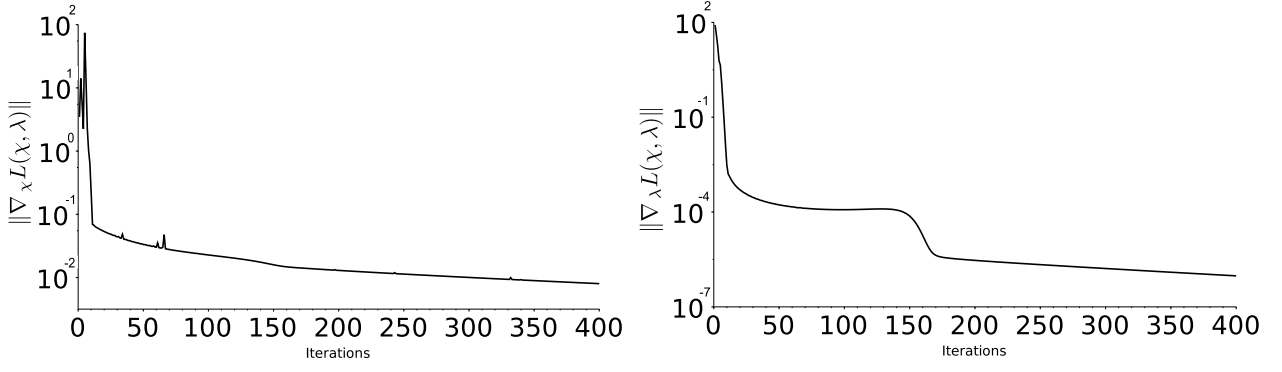


Figure 4: Benchmark 3 with setting  $n = 40$ ,  $N = 20$  and the Hessian approximated by SR1 in trust-region SQP; There are values  $\|\nabla_{\chi} \mathcal{L}(\chi, \lambda)\|$  in the left and  $\|\nabla_{\lambda} \mathcal{L}(\chi, \lambda)\|$  in the right.

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	37	1	20	5	54	1	30	5	46	1	40	5	67	1
	10	47	1		10	54	3		10	122	1		10	57	1
	15	126	1		15	52	1		15	97	1		15	60	1
	20	79	3		20	149	3		20	101	3		20	90	3
	25	185	1		25	107	3		25	80	1		25	98	3
	30	400	2		30	108	1		30	122	1		30	98	1

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	80	1	20	5	110	1	30	5	81	1	40	5	57	1
	10	54	1		10	54	1		10	149	1		10	118	1
	15	152	1		15	126	1		15	111	1		15	56	1
	20	121	1		20	133	1		20	89	1		20	192	1
	25	61	1		25	63	1		25	69	1		25	165	1
	30	152	1		30	68	1		30	178	1		30	142	1

Table 7: Benchmark 3 with the Hessian matrix approximated by BFGS; In the top there are results for line search, and in the bottom there are results for trust region.

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	65	1	20	5	31	1	30	5	49	1	40	5	52	1
	10	68	1		10	F	2		10	37	3		10	51	1
	15	40	3		15	75	1		15	60	1		15	54	1
	20	90	F		20	75	3		20	82	1		20	92	3
	25	21	F		25	F	3		25	101	3		25	99	1
	30	88	1		30	86	1		30	122	1		30	80	1

n	N	NIT	S	n	N	NIT	S	n	N	NIT	S	n	N	NIT	S
10	5	-	F	20	5	400	2	30	5	400	F	40	5	400	F
	10	400	2		10	400	2		10	400	2		10	400	2
	15	400	2		15	400	2		15	400	2		15	400	2
	20	400	2		20	400	2		20	400	2		20	400	2
	25	400	2		25	400	F		25	400	2		25	400	2
	30	400	2		30	400	2		30	400	2		30	400	F

Table 8: Benchmark 3 with the Hessian matrix approximated by SR1; In the top there are results for line search, and in the bottom there are results for trust region.

Lemmas 1 and 2 show the structure of the Hessian  $\nabla_{\chi}^2 \mathcal{L}(\chi, \lambda)$ . Moreover, in the linear case one can use the formulas from Lemma 2 to compute second derivatives fast and using already computed data from the Jacobian of constraints. To a lesser extent the same can be said in the nonlinear case about the result in Lemma 1, where formulas for some elements in the Hessian are provided.

We discussed the spectrum of the Hessian for linear ODEs. In general, we deal with a singular and indefinite upper-left block in the saddle-point system (9). To this end we tried the trust-region method since it may be applied when the Hessian is not SPD and compare it with line search approach.

In our experience the trust-region approach required more iterations than line search. Moreover, as illustrated in Fig. 3 and 4 it is difficult to set a reasonable stopping criterion for the optimality condition  $\nabla_{\chi} \mathcal{L}(\chi, \lambda) = 0$ . Especially when using SR-1 approximations of the Hessian the trust-region method struggles and the norm of the  $\nabla_{\chi} \mathcal{L}(\chi, \lambda)$  stagnates.

We conclude that line search SQP is more appropriate for this problem despite the properties of the true Hessian matrix that suggests the use of trust-region SQP.

## References

- [1] H. Abbas and G. Fainekos. Linear hybrid system falsification through local search. In T. Bultan and P.-A. Hsiung, editors, *Automated Technology for Verification and Analysis*, volume 6996 of *Lecture Notes in Computer Science*, pages 503–510. Springer Berlin Heidelberg, 2011.
- [2] Y. Annpureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan. S-TaLiRo: A tool for temporal logic falsification for hybrid systems. In P. Abdulla and K. Leino, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 6605 of *Lecture Notes in Computer Science*, pages 254–257. Springer Berlin Heidelberg, 2011.
- [3] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. SIAM, 1995.
- [4] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 5 2005.
- [5] M. S. Branicky, M. M. Curtiss, J. Levine, and S. Morgan. Sampling-based planning, control and verification of hybrid systems. *IEEE Proceedings Control Theory and Applications*, 153(5):575, 2006.
- [6] R. Estrin and C. Greif. On nonsingular saddle-point systems with a maximally rank deficient leading block. *SIAM Journal on Matrix Analysis and Applications*, 36(2):367–384, 2015.
- [7] A. Griewank and P. Toint. Partitioned variable metric updates for large structured optimization problems. *Numerische Mathematik*, 39(1):119–137, 1982.
- [8] H. K. Khalil. *Nonlinear Systems; 3rd ed.* Prentice-Hall, Upper Saddle River, NJ, 2002.
- [9] J. Kuřátko and S. Ratschan. Solving Underdetermined Boundary Value Problems by Sequential Quadratic Programming. Submitted and available at <https://arxiv.org/abs/1512.09078>.
- [10] F. Lamiraux, E. Ferré, and E. Vallée. Kinodynamic motion planning: Connecting exploration trees using trajectory optimization methods. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 4, pages 3987–3992. IEEE, 2004.
- [11] L. Lukšan and J. Vlček. Numerical experience with iterative methods for equality constrained nonlinear programming problems. Technical Report V-779, ICS AS CR, 2000.

- [12] L. Lukšan and J. Vlček. Numerical experience with iterative methods for equality constrained nonlinear programming problems. *Optimization Methods and Software*, 16(1–4):257–287, 2001.
- [13] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2nd edition edition, 2006.
- [14] G. Ren-pu and M. J. D. Powell. The convergence of variable metric matrices in unconstrained optimization. *Mathematical Programming*, 27(2):123–143, 1983.
- [15] Scilab Enterprises. *Scilab: Free and Open Source software for Numerical Computation*. Scilab Enterprises, Orsay, France, 2012.
- [16] A. Zutshi, S. Sankaranarayanan, J. V. Deshmukh, and J. Kapinski. A trajectory splicing approach to concretizing counterexamples for hybrid systems. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference*, pages 3918–3925, 2013.